Supply Chain Research · Cornerstone Reference

Supply Chain Software Selection: A Framework

How Do I Choose Supply Chain Software?

A defensible selection rests on four steps: define requirements as observable outcomes and weight them before contacting vendors, screen on disqualifiers, test on your own data, and verify references you sourced yourself. The rule that makes it work is that the weighting is never revised once scores are visible.

CONTENTS

01 The short answer

Choosing supply chain software is a four-step method: define requirements as observable outcomes and fix their weights before any vendor is contacted, screen the long list on hard disqualifiers rather than feature scores, test the short list against scripted scenarios run on your own data, and verify references that you sourced rather than ones the vendor supplied. The discipline that makes the method defensible is simple and frequently broken: the weighting is set once, in writing, and is not revised after scores become visible.

4	3	2
steps, in order, with the weighting fixed at step one	vendors is usually the right short list, not five or six	references minimum that the vendor did not choose

Key takeaways

- **Weight before you look.** Requirements weighted after scores are visible produce justification rather than selection.
- **Write requirements as outcomes, not features.** State what the system must let a named role accomplish, so it can be observed rather than claimed.
- **Screen on disqualifiers, score only the short list.** Detailed scoring across a long list consumes the effort that should go into testing.
- **Test on your data or do not test.** A demonstration on vendor sample data verifies that the vendor can demonstrate, nothing more.
- **Reference the implementation team, not just the product.** Ask who will actually staff your project and check that those names appear in the contract.

02 How should requirements be written and weighted?

Write each requirement as something a named role must be able to accomplish, expressed so that its presence or absence can be observed. A requirement stating that the system supports wave planning cannot be failed, because every candidate will claim it. A requirement stating that a warehouse supervisor can rebuild and release a wave within ten minutes of a carrier cutoff change, without administrator help, can be watched and either happens or does not.

Weighting matters more than the requirement list, and it should be completed and signed before vendor contact. Three tiers are enough: requirements that are mandatory and disqualify a vendor if absent, requirements that carry scored weight, and requirements that are recorded as preferences with no weight at all. Most lists are too long, and the useful test is whether a stakeholder can name what they would give up to get each item. Anything nobody would trade for belongs in the third tier.

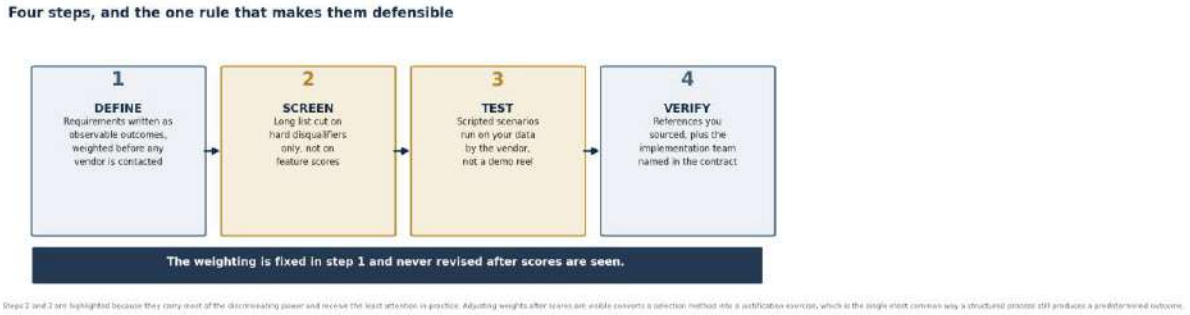

Figure 1. The four steps and the rule that holds them together. Steps two and three carry most of the discriminating power and typically receive the least attention.

03 How do I cut a long list without scoring features?

Use disqualifiers, not scores. At long list stage the objective is to remove candidates that cannot work, and that judgment rests on a small number of structural facts rather than on feature comparison. Typical disqualifiers are deployment model where a constraint exists, industry or regulatory coverage, geographic support for the countries actually operated, integration with a system that is not being replaced, and viable scale, meaning the vendor supports customers of comparable size and complexity.

This is where most processes lose their budget. Scoring forty vendors across two hundred criteria generates a large amount of evidence about how vendors answer questionnaires and almost none about how software behaves. The effort saved by screening structurally is the effort available for scripted testing, which is the step that actually separates products.

The fair case against this is that disqualifiers can exclude a strong candidate on a technicality, particularly where a vendor is weak in a stated area but exceptional overall. That risk is real. The mitigation is to keep the disqualifier list short, require each one to be justified in writing by the person proposing it, and permit a documented override at short list stage.

Stage	What it decides	What it should not do
Define	Which outcomes matter and what each is worth, fixed in writing	Collect every feature any stakeholder can imagine wanting
Screen	Which vendors are structurally viable, using hard disqualifiers	Score a long list in detail against weighted criteria
Test	How each short-listed product behaves on your data and scenarios	Accept a standard demonstration on vendor sample data
Verify	Whether the outcome held for comparable buyers, and who will staff yours	Rely on a reference list assembled by the vendor
Decide	Which product wins on the weights agreed at the start	Revisit the weighting to fit the preferred result

Table 1. The five stages and the failure that shadows each one. The right-hand column describes what usually happens when a selection process produces a predetermined answer.

04 What should a demonstration actually test?

Scripted scenarios on your own data, executed by the vendor in front of the people who will use the system. Supply a data extract, define five or six scenarios drawn from real operating difficulty rather than the happy path, and require the vendor to run them. Good scenarios are the ones that hurt: a short-dated substitution, a split shipment where one line is out of stock, a return of an item bought on a promotion that has since ended, a rush order that arrives after the cutoff.

What to watch is different from what is being presented. Count the clicks and note where the presenter switches user or asks a colleague to take over, because that indicates a role boundary that will become a workflow problem. Ask what happens when a step fails, since exception handling is where most operational time is spent and it is rarely shown voluntarily. Note anything that requires configuration by the vendor rather than an administrator, as that is future dependency and cost.

It is fair to acknowledge the burden. Scripted testing on real data takes effort from both sides, extends timelines, and some vendors resist it. That resistance is itself information. Where a full data extract is impractical, a reduced but real dataset covering the difficult cases still discriminates far better than sample data.

05 How do I check references that mean something?

Insist on at least two references you sourced independently, through industry contacts or user groups, alongside any the vendor supplies. Vendor-supplied references are not worthless, but they are selected, and a selected reference answers a different question than a random one. The most useful reference is a customer of similar size, in a similar industry, who implemented within the last two years and is past the stabilization period.

Ask about the implementation rather than the product. What was in scope at signature and what was in scope at go-live. How many of the named implementation staff stayed for the whole project. What was discovered after go-live that nobody raised during selection. Whether they would buy it again, and separately, whether they would use the same implementation partner. The gap between those last two answers is often the most informative thing in the call.

06 Who should decide, and how?

A small decision group with a named owner, working from the agreed weights, supported by a wider group that contributes requirements and testing observations without holding a vote. Selection by large committee tends toward the option that offends nobody, which is usually the incumbent or the largest vendor, and consensus at that scale is achieved by discounting the sharpest objections rather than resolving them.

Record the decision while the reasoning is still available. A short document stating the weights, the scores, the disqualifiers applied, the overrides granted, and the two or three judgments that determined the outcome is worth writing on the day. It takes an hour and it is the only artifact that will explain, eighteen months later, why the choice was made, which is precisely when someone will ask.

07 Frequently asked questions

How many vendors should be on the short list?

Usually three. Two provides no comparison if one withdraws, and five or more dilutes the testing effort to the point where nothing is examined properly. Three allows scripted testing on real data with each candidate, which is the step that discriminates. Add a fourth only where the market is unsettled enough to warrant it.

Should we hire an advisor to run the selection?

It depends on whether the advisor is independent of the vendors under consideration. Ask directly how they are compensated, whether they hold reseller or implementation relationships with any candidate, and whether their shortlist varies by client. An advisor with implementation revenue from a candidate is not disqualified, but the relationship should be disclosed and weighed.

How long should a selection take?

Long enough to test properly and short enough that the requirements do not go stale. The steps that reward time are scenario design and reference calls; the steps that consume time without adding much are long-list scoring and repeated internal review cycles. SCR does not publish a benchmark duration, because credible public data on selection timelines does not exist.

What if stakeholders disagree on the weighting?

Resolve it before vendor contact, because it becomes far harder afterward. Disagreement at that stage is usually about business priorities rather than software, and it is cheaper to settle in a room than to discover it during scoring. If it cannot be settled, that is a signal the scope itself is unclear.

Can we reuse a weighting from a previous selection?

As a starting point only. Weights encode business priorities at a moment, and the priorities that drove a previous decision may have been superseded. Reusing a structure is sensible; reusing the numbers without review imports assumptions nobody has examined.

Should price be part of the score?

Keep it separate. Blending price into a capability score obscures both, because a small price difference can silently outweigh a large capability gap depending on how the scale was built. Score capability, establish cost separately on a like-for-like scope, and make the trade-off explicitly at the decision.

What if the incumbent is one of the candidates?

Evaluate it on the same terms, and be explicit about the switching cost as a separate line rather than as an adjustment buried in the scoring. Incumbents carry a real advantage that deserves to be counted once, visibly, rather than applied informally at several points in the process.

08 Method, sources, and where to go deeper

Method

- This page describes a constructive method rather than a critique of procurement practice. It draws on the structure of the SCOR Digital Standard for the process boundaries a requirement set should cover, and otherwise reflects SCR's own framing of practitioner method.
- Supply Chain Research is independent and vendor-neutral. We accept no payment from the vendors or categories covered, and this page names no products.

Caveats

- No credible public benchmark exists for selection timelines, short list sizes, or the relationship between process rigor and implementation outcome. The numbers in this page reflect common practice observed across engagements, not measured data, and are presented as guidance rather than evidence.
- Table 1 and Figure 1 are method descriptions rather than research findings. Organizations with mature procurement functions will reasonably run variants of this sequence.
- This page deliberately does not address contract structure, pricing negotiation, or implementation planning, each of which is a separate discipline covered elsewhere in SCR's editorial coverage.

Where to go deeper

This method applies across categories, but the requirements it asks you to write are category specific. The warehouse management, transportation management, order management, and supply chain planning guides each set out the capability areas and evaluation criteria that matter in those categories, and are the right place to source the requirement list this framework then weights and tests. Readers scoping across several categories at once should start with the SCR supply chain software category map.

Sources

[1] Association for Supply Chain Management. SCOR Digital Standard overview.

[2] Association for Supply Chain Management. SCOR DS model reference, process definitions used to scope requirement coverage.

[3] Association for Supply Chain Management. Introduction and front matter, SCOR Digital Standard, 2025 edition.